



Network Coding to Enhance Standard Routing Protocols in Wireless Mesh Networks

Pahlevani, Peyman; Roetter, Daniel Enrique Lucani; Fitzek, Frank; Pedersen, Morten Videbæk

Published in:
2013 IEEE Globecom Workshops (GC Wkshps)

DOI (link to publication from Publisher):
[10.1109/GLOCOMW.2013.6825055](https://doi.org/10.1109/GLOCOMW.2013.6825055)

Publication date:
2013

Document Version
Early version, also known as pre-print

[Link to publication from Aalborg University](#)

Citation for published version (APA):
Pahlevani, P., Roetter, D. E. L., Fitzek, F., & Pedersen, M. V. (2013). Network Coding to Enhance Standard Routing Protocols in Wireless Mesh Networks. In *2013 IEEE Globecom Workshops (GC Wkshps)* (pp. 610-616). IEEE Press. <https://doi.org/10.1109/GLOCOMW.2013.6825055>

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

Take down policy

If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Network Coding to Enhance Standard Routing Protocols in Wireless Mesh Networks

Peyman Pahlevani, Daniel E. Lucani, Morten V. Pedersen, Frank H.P. Fitzek
Aalborg University, Department of Electronic Systems
Email: {pep|del|mv|ff}@es.aau.dk

Abstract—This paper introduces a design and simulation of a locally optimized network coding protocol, called PlayNCool, for wireless mesh networks. PlayNCool is easy to implement and compatible with existing routing protocols and devices. This allows the system to gain from network coding capabilities implemented in software without the need for new hardware. PlayNCool enhances performance by (i) choosing a local helper between nodes in the path to strengthen the quality of each link, (ii) using local information to decide when and how many transmissions to allow from the helper, and (iii) using random linear network coding to increase the usefulness of each transmission from the helpers. This paper focuses on the design details needed to make the system operate in reality and evaluating performance using ns-3 in multi-hop topologies. Our results show that the PlayNCool protocol increases the end-to-end throughput by more than two-fold and up to four-fold in our settings.

I. INTRODUCTION

Traditional routing protocols in wireless mesh networks use a single route to deliver packets from the source to the destination as shown in Fig. 1(a), where the source (S) established a single path to destination (D) using relays R_1 and R_2 [1]–[3]. The route is established by finding the best next hop for a newly arrived packet based on different criteria (e.g., number of hops, delay). More recently, multi path routing protocols in wireless networks rely on sharing the load between different paths [4]. This approach is similar to the routing protocols for wired networks and does not exploit the broadcast nature of the wireless network. However, the broadcast property of the wireless medium enables the nodes to overhear packets transmitted to other nodes. Managing packet overhearing can open the door to an increase in the reliability and the throughput in wireless networks.

In contrast to traditional routing protocols in wireless

This work was partially financed by the Green Mobile Cloud project granted by the Danish Council for Independent Research (Grant No. 10-081621).

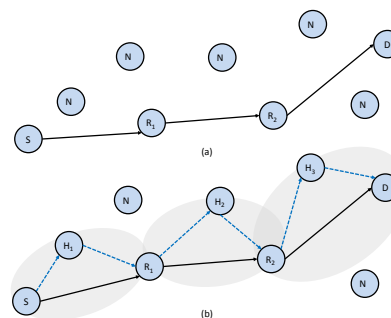


Fig. 1. (a) Traditional routing. (b) PlayNCool helper approach. Grey areas illustrate local optimization with one helper for a link.

networks, the ExOR protocol [5] is an integrated routing and MAC technique that utilizes the cooperation between nodes using the broadcast nature of the wireless medium. However, the nodes in ExOR need to coordinate their actions to avoid transmitting duplicate packets. In order to decrease the coordination overhead between nodes, MORE [6], CCACK [7], and GeoCode [8] have exploited random linear network coding (RLNC) [9], a technique that allows the sender to transmit random linear combinations of the original packets. MORE is an opportunistic routing protocol that implements RLNC. Each relay will store coded packets for an active generation and recode before sending, i.e., creating new linear combinations based on the contents of its buffer of coded packets. Although MORE is interesting, its impact on commercial systems may be on the longer term rather than a short or medium term because it lacks compatibility to standard routing protocols and it relies on an offline calculation of the error probability of each link. GeoCode [8] creates multiple paths by choosing the nodes that are located inside a specified geographic area (e.g. ellipse) as relay nodes. The created paths may intersect each other at intermediate nodes which use network coding to maximize the throughput. CCACK [7] uses a Cumulative Coded acknowledgement approach to acknowledge network

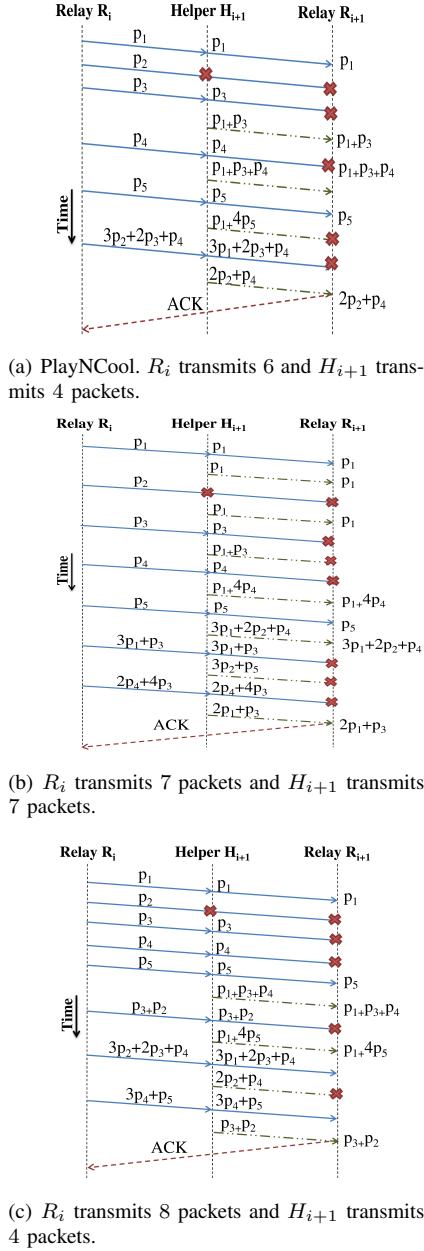


Fig. 2. R_i transmits coded packets to R_{i+1} using a helper. The helper is activated when it has received (a) two packets (b) one packets (c) 4 packets.

coded flow to their upstream nodes, where the feedback is resilient to packet losses.

In contrast with this paradigm, where traditional routing protocols are essentially changed for novel protocols, we focus on PlayNCool, a network coding protocol that aims at having impact in the short term by providing significant performance enhancements to existing routing protocols in an easily implementable way. To achieve this compatibility, PlayNCool chooses a

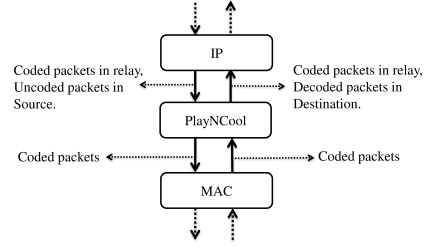


Fig. 3. Protocol stack with PlayNCool. The source IP layer delivers the uncoded packets to the PlayNCool layer. The PlayNCool layer codes them and puts them in the MAC layer. The relay receives the coded packets in MAC layer and delivers them to PlayNCool and from there to the IP layer.

local helper between nodes in the path in order to strengthen the quality of each individual link in that path (as shown in Fig. 1(b)). The key parameter to guarantee the performance of PlayNCool is that each helper should *play it cool* and wait until it has received enough packets before it starts to transmit. A distinctive feature of PlayNCool is that it is aware to competition from other nodes. Whenever the network becomes congested due to other flows, PlayNCool can activate the helper sooner to increase the priority of the current flow. PlayNCool relies on local knowledge of the channel conditions to determine when and how much to transmit. In particular, it uses link quality information either provided by routing protocols, e.g., B.A.T.M.A.N. [3], or by estimating it based on the transmitted and received packets locally and calculated at the PlayNCool layer. In our design, we inserted the PlayNCool layer to the network stack between IP and MAC as shown in Fig. 3. Our simulations in ns-3 considering deployments of up to 25 nodes, show a performance improvement of two to four fold compared to traditional routing.

II. GENERAL FRAMEWORK OF THE PLAYNCool PROTOCOL

The basic topology of PlayNCool, including relays and a helper, is shown in Fig. 4. The relay (R_i) sends the coded packets to the next relay (R_{i+1}). The helper (H_{i+1}) overhears the coded packets from R_i . We define R_0 as the source and R_n as the destination for an n -hop network. This figure illustrates the error probabilities of the different links. The error probabilities between R_i and H_{i+1} , H_{i+1} and R_{i+1} , and R_i and R_{i+1} are represented by $e_1^{(i+1)}$, $e_2^{(i+1)}$, and $e_3^{(i+1)}$, respectively. The error probabilities are assumed to be available for R_i and R_{i+1} . In this section, we describe the actions of PlayNCool taken by each node upon receiving a new generation.

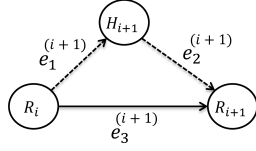


Fig. 4. PlayNCool basic topology.

• General actions of the nodes

The network layer of the source (S) in Fig. 1(b) delivers uncoded packets to the PlayNCool layer. The PlayNCool layer then divides the packets into batches of g packets (each batch is called a generation). For each generation, the source generates coded packets using random linear network coding, then it stores them in the MAC layer queue. Once the coded packets are buffered in the MAC queue for transmission, they cannot be removed from the MAC queue because we design the PlayNCool protocol to be independent from the upper and the lower layers. The source transmits the coded packets buffered in the MAC queue to the relay (R_i). It can choose a helper to increase the reliability of its link to the relay, e.g., sending a request to a possible helper node. Alternatively, a helper can start transmitting overheard packets spontaneously when the link quality between source and relay is low. The helper (H_i) in Fig. 1(b) accumulates the coded packets by overhearing transmissions from the source. When it accumulates a number of coded packets (less than g in general), it generates coded packets by recoding, i.e., by creating linear combinations of the buffered coded packets, and transmits them to the intended relay. At this point, both the source and the helper continue to transmit coded packets to the relay until the relay signals that it has all g . Then, the source stops transmitting the current generation and starts transmitting the new generation. Each helper controls the number of the coded packets that should be generated by a metric called *Budget*.

The relay starts to recode and generate coded packets upon reception of packets from S and H_i , and transmits them to the next relay. In the following, we will describe the detailed actions taken by each node upon receiving a generation.

• Source actions

1) *Estimating the number of coded packets that need to be transmitted:* First, the source estimates the number of coded packets that need to be transmitted on the link. The estimation is based on a heuristic mean value analysis of the DOF delivered, considering the loss probabilities of the links. The key parameter

to determine is the number of overheard packets in the helper before it starts transmitting the coded packets (called p). The value of p will determine how much the source and the helper should send, but it also determines the performance of the scheme.

The value of p should be large enough to guarantee that the helper transmissions are innovative for the relay with high probability. If p is too small, the helper starts transmitting too early and the relay will receive linearly dependent packets. Therefore, some of the early transmissions will be wasted. If p is too large, the helper starts transmitting too late, which means the relay may have received most of the DOFs from the source and the usefulness of the helper is limited. Let us consider the examples of Fig. 2. In Fig. 2(b), choosing $p = 1$ results in the relay R_{i+1} receiving two non-innovative coded packets from the helper H_{i+1} . Thus, the total number of transmission of packets is 14. In Fig. 2(c), the helper starts transmitting when it has received $p = 3$ coded packets. In this case, the total number of transmissions is 12. On the other hand, if we choose $p = 2$ only 10 transmissions are needed, as shown in Fig. 2(a).

The total number of transmitted packets from the source is equal to the number of transmissions before the helper is activated (r) plus the number of transmissions after the helper is activated (k). By considering the error probability between the source and the helper, p is given as $p = (1 - e_1^{(i+1)}) \cdot r$. In our analysis, we assumed that MAC shares the channel between nodes equally, which is a valid assumption for CSMA/CA. Based on this assumption, the helper and the source transmit k coded packets after the helper is activated. We also assumed that the field size of RLNC is large enough so that the probability of receiving linearly dependent coded packets is low. In order to determine p , as well as the number of transmissions from source and helper, we proceed with the following mean flow analysis. We consider that the relay should receive g innovative coded packets in total from the helper and the source to decode a generation. Thus:

$$g = r \cdot (1 - e_3^{(i+1)}) + k \cdot (1 - e_2^{(i+1)}) + k \cdot (1 - e_3^{(i+1)}) \quad (1)$$

We calculate r for two cases. In the first, the number of incoming innovative packets to the helper is higher than the number of outgoing packets from the helper, i.e., $(1 - e_1^{(i+1)}) \cdot e_3^{(i+1)} > 1 - e_2^{(i+1)}$. Thus, the helper starts to transmit upon receiving the first innovative packet. The number of transmissions

until the helper receives an innovative packet is $r_a = \frac{1}{(1-e_1^{(i+1)}) \cdot e_3^{(i+1)}}$, which means that $p = \frac{1}{e_3^{(i+1)}}$.

In the second case $(1 - e_1^{(i+1)}) \cdot e_3^{(i+1)} \leq 1 - e_2^{(i+1)}$. Considering that the number of incoming innovative packets in the helper should be equal to the number of outgoing innovative packets from the helpers, then $r_b \cdot (1 - e_1^{(i+1)}) \cdot e_3^{(i+1)} + k \cdot (1 - e_1^{(i+1)}) \cdot e_3^{(i+1)} = k \cdot (1 - e_2^{(i+1)})$.

Combining Eq. (1) and equation above, allows us to calculate

$$r_b = \frac{-g \cdot L(e_1^{(i+1)}, e_2^{(i+1)}, e_3^{(i+1)})}{G(e_1^{(i+1)}, e_2^{(i+1)}, e_3^{(i+1)})}. \quad (2)$$

where $L(a, b, c) = -1 + b + c - a \cdot c$ and $G(a, b, c) = (2 - c - b) \cdot (c - a \cdot c) - (1 - c) \cdot L(a, b, c)$.

The number of coded packets that need to be transmitted on the link from the source is:

$$B_s(r) = \frac{g + (1 - e_2^{(i+1)}) \cdot r}{2 - e_3^{(i+1)} - e_2^{(i+1)}}. \quad (3)$$

where r is r_a (r_b) for case 1 (2).

2) *Generating coded packets*: The source generates random linear network coded packets and put them in the MAC layer. The number of coded packets is calculated by Eq. (3).

3) *Generating additional coded packets*: As described above, the source estimates the number of coded packets that need to be transmitted. However, due to randomness in the channel and dynamic characteristic of the wireless networks, this estimate, which is based on mean performance, does not guarantee generation of enough coded packets. PlayNCool solves this problem using one of two alternatives. (I) monitoring the transmission queue or by (II) estimating the completion time.

(I) The queue monitoring mechanism monitors the transmission queue to guarantee that there is enough number of packets in the transmission queue to transmit. When the number of coded packets in the queue is less than a certain threshold (e.g., 10 coded packets) and the source has not received any acknowledgement (ACK) packet, the source generates extra coded packets of the generation (e.g., 10% extra). (II) Expected estimating timeout for transmitting the generation allows the source to send if no ACK is received after some time. If the transmission time of a packet is t , the timeout of a generation is $t \cdot (r + k)$. When the timeout of a generation is elapsed and the source

has not received any ACK yet, it generates 10% extra coded packets from that generation. A new timeout is set for the additional coded packets. The process is repeated until an ACK is received.

4) *Stop transmission of the current generation*: Finally, the source stops transmitting the current generation and starts transmitting the next generation upon receiving an ACK for the current generation.

• Helper actions

1) *Accumulating enough coded packets before start transmitting*: As we discussed before, the helper starts transmitting when it has accumulated p coded packets.

2) *Estimating the number of coded packets that need to be transmitted*: The helper should transmit k coded packets when it is activated. It calculates k from Eq. (1) as the following:

$$k(r) = \frac{g - r \cdot (1 - e_3^{(i+1)})}{2 - e_3^{(i+1)} - e_2^{(i+1)}}. \quad (4)$$

where r is calculated for different cases (r_a and r_b).

3) *Increasing the budget by overhearing the source packets*: The helper should not generate all the k coded packets at the same time. This will increase the number of non-innovative transmissions because the helper should not generate much more than it knows, e.g., it should not generate 50 coded packets if it has received only one coded packet. Each helper H_{i+1} controls the number of the coded packets that should be generated by a metric called *budget* $B_h(t + 1)$. Budget $B_h(t + 1)$ is the number of coded packets that the helper can generate at time $t + 1$:

$$B_h(t + 1) = B_h(t) + C_h^{(i+1)} - N_h^{(i+1)}(t) \quad (5)$$

Where the credit $C_h^{(i+1)}$ is the number of generated coded packets in the helper H_{i+1} per a new incoming coded packet and $N_h^{(i+1)}(t)$ is the number of transmitted packets at time t . $N_h(t)$ can be increased until $B_h(t + 1) \in [0, 1)$.

As shown in Fig. 4, when the source transmits w coded packets, the helper receive $y = w \cdot (1 - e_1^{(i+1)})$ coded packets in the expectation. Consequently, when a helper receives one coded packet ($y = 1$) from the source it increases its budget by a credit value equal to:

$$C_h^{(i+1)} = (1 - e_1^{(i+1)})^{-1} \quad (6)$$

4) *Generating coded packets*: When the budget is higher than one, the helper generates a recoded packet and decreases the budget by one. Then, it puts the recoded packets in the transmission queue. When the

budget reaches zero, it stops generating the packets until the budget is increased again.

5) *Terminating the transmission of the coded packets:* The helper finishes transmitting coded packets either when it has received an ACK packet from the relay or when it has finished generating all k coded packets in Eq. (4).

• Relay actions

1) *Increasing the budget by receiving a new packet:* Similar to the helper, a relay should not transmit much more than it knows, to avoid non-innovative packet transmissions. The relay controls the number of generated packets by the relay budget $B_r(t+1)$.

$$B_r(t+1) = B_r(t) + C_r^{(i+1)} - N_r^{(i+1)}(t) \quad (7)$$

where the credit $C_r^{(i+1)}$ is the number of generated coded packet in relay r_{i+1} per new incoming coded packet and $N_r^{(i+1)}(t)$ is the number of transmitted packets at time t .

When the relay R_i transmits a coded packet, the helper H_{i+1} and the relay R_{i+1} both can increase the number of DOF. Therefore, the number of coded packets needs to be transmitted from the R_{i+1} until the helper or the relay receives a new coded packet is equal to:

$$C_r^{(i+1)} = \left(1 - e_3^{(i+1)} \cdot e_1^{(i+1)}\right)^{-1} \quad (8)$$

2) *Generating coded packets:* The relay generates a recoded packet when the budget is higher than one.

3) *Transmitting an ACK packet to the previous relay (or source):* The relay transmits an ACK packet to the relay R_i when it has received g DOF.

4) *Generating additional coded packets:* To be sure that there are always enough coded packets in the transmission queue, the relay generates additional coded packets following one of the strategies of the source.

5) *Stop generating coded packets:* The relay stops transmissions for the current generation and starts generating packets for the next after receiving an ACK from the next relay.

Remarks

Different policies for activating the helper are possible. Two policies are described. The calculation of credits and budget follows the same ideas as for PlayNCool.

1) *WFF: Wait for all of degrees of freedom policy:* The helper starts transmitting when it has accumulated all DOF. By using the same analysis from previous

section the transmission budget for the relay is equal to Eq. (7). Since the helper does not need to overhear the coded packets after it starts transmitting, the budget value does not increase by overhearing and it is fixed. By using the same analysis from the PlayNCool, the transmission budget in the helper is given as:

$$B_h = \frac{g - r \cdot \left(1 - e_3^{(i+1)}\right)}{2 - e_3^{(i+1)} - e_2^{(i+1)}}, \quad (9)$$

Which, $r = \frac{g}{1-e_1}$. The value of the budget in the helper will be decreased by one when it generates a coded packet until it reaches zero. Then, the helper will not generate any more packets.

2) *WFH: Wait for a fraction of the degrees of freedom policy:* The helper starts transmitting when the helper and the relay receive enough DOF jointly to decode a generation. Then, the source stops transmitting the coded packets.

By using the same type of the analysis from PlayNCool, the budget in the relay is equal to Eq. (7). When the helper starts transmitting, the source will not transmit any coded packets. Therefore, the budget value will not increase by receiving a new coded packet. Thus, the budget is same as Eq. (9), where, $r = g \cdot \left(1 - (e_1^{(i+1)} \cdot e_3^{(i+1)})\right)^{-1}$.

III. IMPLEMENTATION DETAILS IN NS-3

In order to verify the performance of the PlayNCool policy, we used the ns-3 simulator. ns-3 is an open source discrete event network simulator intended to be used for research on computer networks. In our implementation, the source sends a UDP flow to the destination using static routing protocol in IP layer. Moreover, the IEEE 802.11b [10] standard is used. In this implementation, for simplicity, we have chosen a predefined helper between two relays.

• *The interaction of PlayNCool with the other layers:* As shown in Fig. 3, we inserted the PlayNCool layer between the MAC layer and IP layer in the ns-3 protocol stack. The PlayNCool layer in the source receives uncoded packet from the IP layer and after coding the packets it delivers them to the MAC layer. When the relay receives the coded packets, it forwards them from the MAC layer to the IP layer. Once the IP layer finds the next hop for the coded packets, it returns the coded packets to PlayNCool. The PlayNCool layer decodes the packets and delivers them to the MAC layer. The IP layer of the helper does not need to find the next hop for the overheard packets because it knows the sender and receiver from

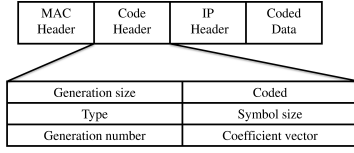


Fig. 5. PlayNCoool packet header format.

the packet header. Thus, it should not forward the coded packets to the IP layer. The PlayNCoool layer overheard the code packets from R_i and it recodes and forwards them to R_{i+1} .

- *PlayNCoool packet format*: When the source generates a coded packet, it adds an IP header and a PlayNCoool header for each coded packet as it shown in Fig. 5. The IP header cannot be coded because the IP layer needs IP header to find the next hop. Each relay uses PlayNCoool header to recode packets of a generation. The PlayNCoool header includes: The *generation size* field indicates the size of that generation, the *symbol size* field indicates the size of each coded packets, the *coded* flag indicates that the packet is coded or uncoded, the *type* field indicates the ACK packets, the *generation number* field indicates the generation number of packets, and the *coefficient vector* represents the coefficient vector of the coded packets.

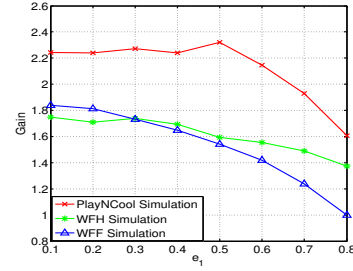
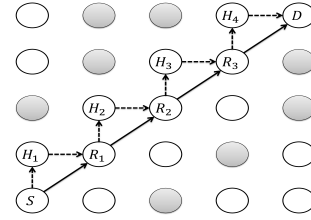
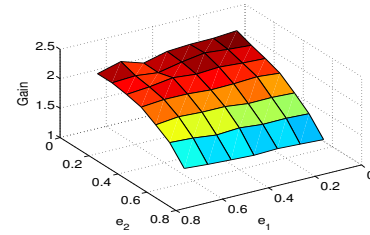
- *Network Coding Library*: In order to code and decode the packets Kodo library [11] is used. Kodo is a network coding library, used for commercial software and for research.

IV. PERFORMANCE COMPARISON AND SIMULATIONS

In this section, we compare the gain of PlayNCoool approach discussed in the previous section. Gain is defined as the completion time when there is no helper node divided by the completion time of different helper policies.

- *The gain of PlayNCoool for basic topology*: First we compare the simulation result of PlayNCoool approach with the other helper policies in the basic topology for single hop. This basic topology for three nodes includes a sender, a helper, and a receiver as Fig. 4. We define x as the number of neighboring nodes that generate extra load in the network. In this network we have $g = 50$, $e_2^{(i+1)} = 0.6$, $e_3^{(i+1)} = 0.8$, and $x = 5\%$. Fig. 6 compares the simulation result when $e_1^{(i+1)}$ is changing.

The PlayNCoool highest gain is close to 2.2. This means that the PlayNCoool protocol transmits 2.2 times faster than approach without helper. This gain is much


 Fig. 6. Simulation for different approaches. $e_3^{(i+1)} = 0.8$, $g = 50$, $x = 5$.

 Fig. 7. The meshed topology including 25 nodes. The grey nodes are generation load in the network. The source sends 12 generation to destination using PlayNCoool. $g = 50$, $e_3^{(i+1)} = 0.8$.

 Fig. 8. Gain of PlayNCoool approach for different link quality in 5×5 mesh network. $e_3^{(i+1)} = 0.8$, $g = 50$, and 8 nodes generates extra packets with 100KBps rate.

higher than the gain of two other policies. It shows that once $e_1^{(i+1)}$ becomes harsh and reaches to 0.8, there is still a high gain close to 1.5.

- *The gain of PlayNCoool for mesh network*: The performance of the end to end transmission with different link quality conditions is interesting metric for the wireless networks. Due to that, a grid of 25 nodes in the wireless mesh networks is defined in Fig. 7. The source (S) transmits 12 generations to the destination (D) through three relays. The gray nodes generate extra traffic in the network to increase the competition between nodes to access the channel. In this topology, we define $g = 50$ and $e_3^{(i+1)} = 0.8$. As shown in Fig. 8, the gain is mostly determined by $e_2^{(i+1)}$ and it reaches to 2.3. However, when $e_2^{(i+1)} = 0.6$ the gain is close to 1.5 which is quite high considering the high losses on packets coming from the helper.

- *The effect of the load on the PlayNCoool gain*: The

competition between nodes to access the channel has a significant effect in the gain. By increasing the load, the competition between nodes will be increased. To study the effect of the load, we assumed a grid with 25 nodes as it shown in Fig. 7. All the nodes around the main flow are generating the load.

Fig. 9 shows the result when $e_1^{(i+1)}=0.4$, $e_2^{(i+1)}=0.4$, $e_3^{(i+1)}=0.8$, and the transmission rate of the nodes is varies. The gain of the PlayNCool protocol increases by increasing the transmission rate and it reaches to 4 in the highest point. The reason PlayNCool has high gain is because having the source and the helper active at the same time allows them to access the channel more frequently and to increase the priority of the current flow for a fraction of the total transmission time in PlayNCool. Given that this reduces the total transmission time quite dramatically, this temporary increase of transmissions has a positive effect in the overall system. When the rate reaches to the highest point, the gain of the PlayNCool protocol will be stabilized because the MAC protocol shares the channel between nodes equally and the channel is fully congested. Fig. 10 studies the effect of the $e_2^{(i+1)}$ loss probability and the load in the gain. When the load in the network is low, by increasing the error probability the gain will be decreased. On the contrary, the effect of the $e_2^{(i+1)}$ on the gain will be minor in the high load because the collision rate is higher than the loss rate and the gain determined mostly by the load.

V. CONCLUSION

In this paper a market oriented optimization for meshed networks using network coding in a localized set up is presented. This protocol, called PlayNCool, exploits local helpers to increase the gain and throughput from the source to the destination and it is compatible with existing routing protocol. The ns-3 implementation showed that PlayNCool increases the end-to-end gain by factor of two to four fold in the wireless mesh network. In this paper a market oriented optimization for meshed networks using network coding in a localized set up is presented. This protocol, called PlayNCool, exploits local helpers to increase the gain and throughput from the source to the destination and it is compatible with existing routing protocol. The ns-3 implementation showed that PlayNCool increases the end-to-end gain by factor of two to four fold in the wireless mesh network. Our future work will focus on the implementation aspects of PlayNCool with the B.A.T.M.A.N routing protocol. The B.A.T.M.A.N routing protocol provides allowing for a straightforward implementation of PlayNCool

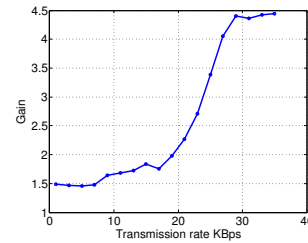


Fig. 9. The effect of the load to the gain in PlayNCool. $e_1^{(i+1)}=0.4$, $e_2^{(i+1)}=0.4$, $e_3^{(i+1)}=0.8$, $g=50$.

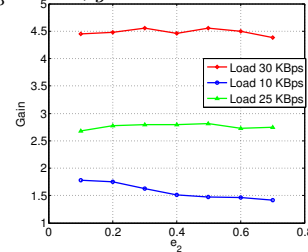


Fig. 10. The effect of the load and error probability of the helper and the relay (R_{i+1}) to the gain in PlayNCool. $e_1^{(i+1)}=0.3$, $e_3^{(i+1)}=0.8$, $g=50$.

and simplifying the selection of helpers based on results presented on our current paper.

REFERENCES

- [1] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks," in *Ad Hoc Networking*, Ed. C. E. Perkins. Addison-Wesley, 2001, pp. 139–172.
- [2] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *proceedings of the 2nd ieee workshop on mobile computing systems and applications*, 1997, pp. 90–100.
- [3] A. Neumann, C. Aichele, M. Lindner, and S. Wunderlich, "Better approach to mobile ad-hoc networking," in *IETF Int. Draft*.
- [4] Y. Yuan, H. Chen, and M. Jia, "An optimized ad-hoc on-demand multipath distance vector (AOMDV) routing protocol," in *Asia-Pacific Conf. on Communications*, 2005.
- [5] S. Biswas and R. Morris, "ExOR: Opportunistic multi-hop routing for wireless networks," in *SIGCOMM*, 2005.
- [6] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "MORE: A network coding approach to opportunistic routing," in *SIGCOMM*, 2007.
- [7] D. Koutsonikolas, C.-C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *IEEE INFOCOM*, 2010.
- [8] H. Khamfroush, D. Lucani, and J. Barros, "GeoCode: A geographic coding-aware communication protocol," in *International IEEE Conf. on Intel. Transp. Sys. (ITSC)*, 2011.
- [9] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. on Inf. Theory*, vol. 52, no. 10, 2006.
- [10] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE Standard 802.11, 1999.
- [11] M. V. Pedersen, J. Heide, and F. H. Fitzek, "KODO: An open and research oriented network coding library," in *Workshop on Network Coding App. and Pro. (NC-Pro)*, Spain.